PLANNING FOR THE EVOLUTION OF AUTOMATED TOOLS IN HLA

Ken Hunt
AEgis Research Corporation
6703 Odyssey Drive, Suite 200
Huntsville, AL 35806
khunt@aegisrc.com

Dr. Judith Dahmann
DMSO
1901 N. Beauregard St., Suite 504
Alexandria, VA 22311-1705
jdahmann@triton.dmso.mil

Robert Lutz
John Hopkins University Applied Physics Laboratory
Bldg 13 - North, Room 414
11100 John Hopkins Road
Laurel, MD 20723-6099

Jack Sheehan
DMSO
1901 N. Beauregard St., Suite 504
Alexandria, VA 22311-1705
jsheehan@msis.dmso.mil

KEYWORDS
Automation, Tools, OMDT, FOM, SOM, DIF

ABSTRACT

A Federation Development and Execution Process (FEDEP) has been developed based on the experiences of a set of prototype HLA federations. Automated software tools have been broadly discussed as applicable to this process. The Object Model Development Tool (OMDT) is the first instance of such a tool. The purpose of this paper is to extract a high-level definition of a set of automated tools from the FEDEP and to outline a strategy for evolving this definition in such a way as to insure that the tools are extensible, open and interoperable. Such a strategy is recognized as crucial to insure that the maximum benefit will be captured from the investments made by end users of the tools in developing HLA-compliant simulations and federations.

1.0 GOALS FOR A TOOL SUPPORT STRATEGY

The anticipated benefits of HLA to the modeling and simulation community in terms of increased software reusability and reduction in overall maintenance are quite significant. However, early experiences with HLA have brought to focus a need for automated tools for creating, executing and maintaining HLA simulations and federations. The modeling and simulation community which constitutes the end-users of HLA certainly possesses the technical expertise and resources to fill this need. However, without an HLA-wide vision for a tool architecture, and a plan for evolving the prototype tools already under development to fit into that architecture, the maximum potential benefits that HLA can provide would not be met.

To develop our strategy for supporting the evolution of automated tools, we begin by identifying the goals of such a plan. First, we desire that a tool support strategy be made manifest as an *open specification*. By open, we mean that the technical details of data interfaces are formally specified and publicly available. The intent of an open tool architecture is to promote the development of a variety of tools applicable to the various activities associated with HLA simulations and federations, so that end-users have the freedom to choose among a selection of existing tools, or even implement their own custom solutions.

Our second goal is to establish a tool support strategy that *promotes interoperability*. We desire to achieve interoperability not only among the tools that we can currently identify, but also among tools that will be conceptualized by future users of HLA.

Our third goal is to develop a strategy for tools that *can be evolved* as the needs of HLA users grow. Although the protofederations and early experiments with HLA have provided a broad set of experiences with which to evolve the current view of the FEDEP, this process is itself in a very early stage. As HLA is adopted by the modeling and simulation community at large, the FEDEP definition is certainly expected to evolve and mature to reflect the practices of the broadened set of users. Likewise, the classifications and implementations of automated tools will evolve, driven by changes to the FEDEP as well as advances in software technology.

2.0 TOOL ARCHITECTURE COMPONENTS

Before developing the strategy for supporting tools in HLA, we will first develop an overview of the set of components that would compose an HLA tool architecture. By architecture components we mean: automated software tools, formal archives or repositories of data, and certain federation run-time components. The federation run-time components that we consider part of a tool architecture are applications that exist as federates, but exi st to assist with controlling, monitoring or providing some assistance other than modeling of scenario objects during the execution of a federation. We will develop our set of architecture components by examining the FEDEP and identifying labor/data intensive areas that could benefit from the application of automation. We will then use this tool architecture as the baseline that our tool support strategy must be supportive of.

Our major criterion for inclusion in the tool architecture will be that we assess the component to be reusable across a variety, if not all, application domains. One-of-a-kind components designed for specific federations are not of particular interest in our plan to support a tool architecture, but are certainly not excluded in terms of interoperability. Only components that have a degree of reusability and can help users capitalize on the efforts of previous federations by either providing access to reusable information or by assisting with steps in a common process are specified.

2.1 Overview of the FEDEP

The FEDEP is described by Figure 1. A more detailed explanation is available through the online HLA Technical Library under the title "HLA Federation Development and Execution Process Model" at http://www.dmso.mil.

For the purpose of examining the FEDEP for automation opportunities, we will decompose the FEDEP into five phases: Design, Development, Testing, Execution and Analysis. The Design phase consists of these FEDEP activities:

- Federation Sponsorship
- Conceptual Analysis
- Scenario Development
- Federation Design

The Development phase consists of:

- FOM Development

- Federation Development

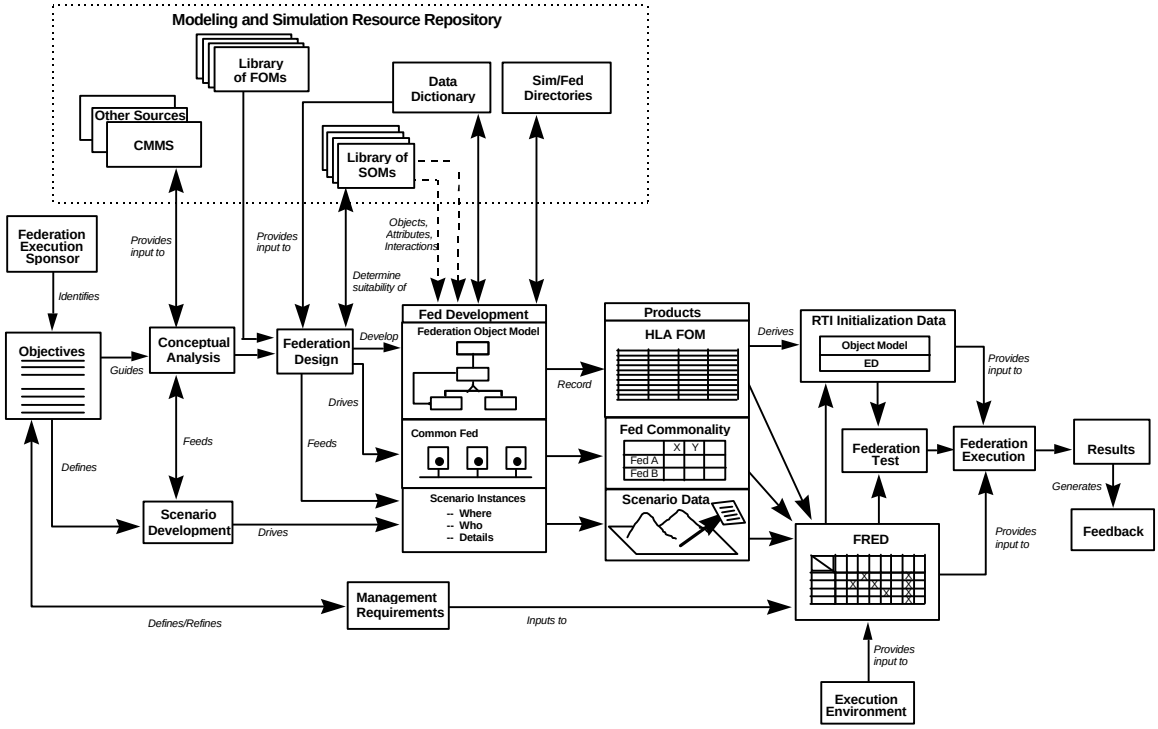

Figure 1. HLA FEDEP Model

The Testing phase consists of:

- Federation Testing

The Execution phase consists of:

- Federation Execution

The Analysis phase consists of:

- Results
- Feedback

The steps associated with these activities will be used to identify components in the architecture.

2.2 Summary of Components

The following components of a tool architecture are either under prototype development or have been recognized as needed due to the HLA prototype federation experiences. Though some of the components have applicability across more than one of the phases of the FEDEP, we have categorized each component with the phase it is most commonly associated with. The development status referred to in each architecture component description refers to DMSO-initiated efforts only. Figure 2 provides a graphical overview of these components and their relationships.

2.2.1 Design Phase Components

1. Conceptual Analysis/Federation Design Support Tool - This tool provides an interface to the conceptual models of mission space (CMMS), the Object Model Library, and the Simulation/Federation Directories. Its primary purpose is to allow federation developers to develop a conceptual view of the objects and interactions that must be supported by the federation, and then form a federation which supports the conceptual analysis. Preliminary studies of this tool have been initiated.
2. Scenario Development Tool - Provides a means to define scenario domain objects and their relationships (affiliation, C^3 hierarchies, etc.), event timelines and initial state information. Due to the extreme diversity of simulations and their requirements for scenario specification, it may well be that no single tool is capable of supporting all HLA federations, although it is likely that classe s of scenario development tools may evolve that satisfy

particular classes of simulations. No development has been initiated for such a tool within the HLA program

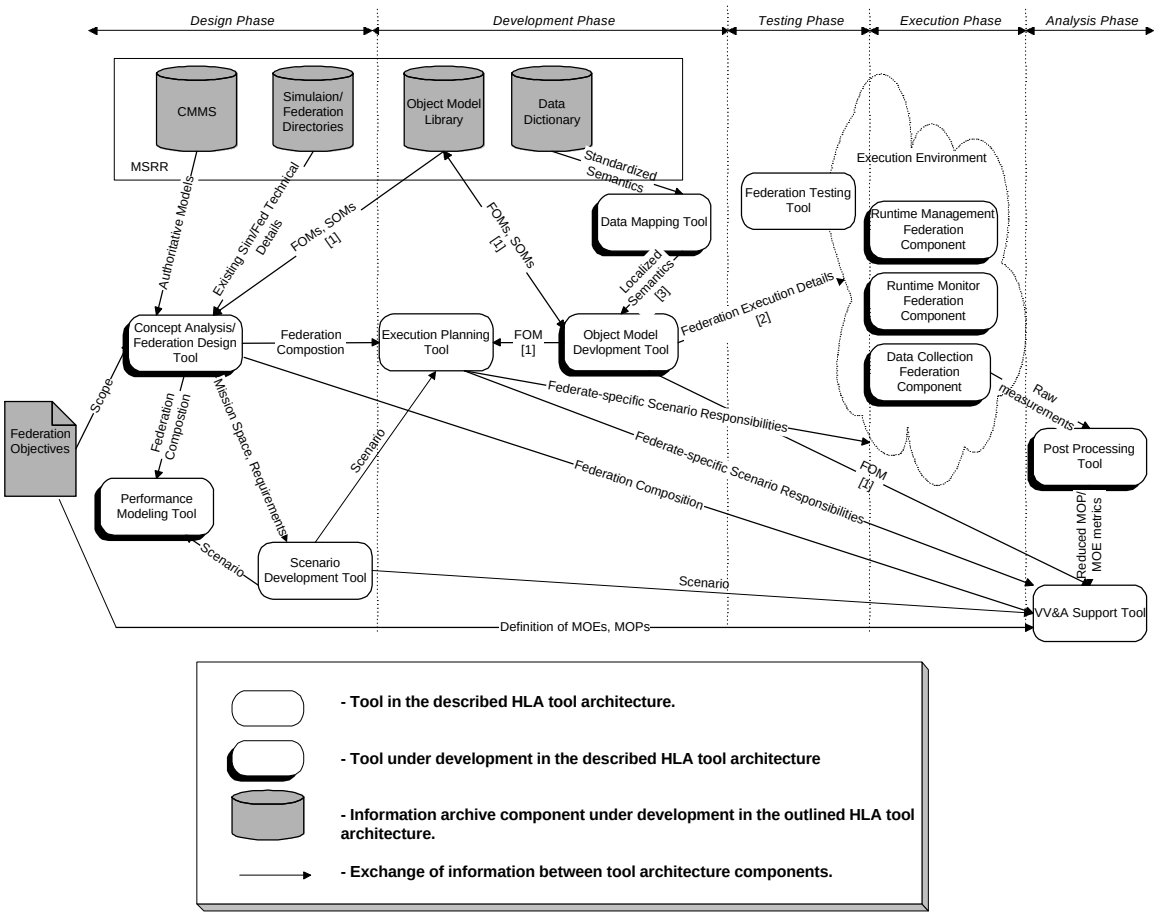

Figure 2. HLA Tool Architecture Components

3. Performance Modeling/Prediction Tool - Based on the execution environment, time management scheme and number of objects being modeled, this tool uses stochastic models of the federates and the RTI to predict a rough estimate of overall federation performance. This tool is currently under development.

4. Simulation/Federation Directories - These components of the Modeling and Simulation Resource Repository contain pointers to full technical descriptions of each DoD simulation and federation. This component is under development.

2.2.2 Development Phase Components

1. Object Model Development Tool (OMDTs) - Automates the data entry and consistency checking needed to specify simulation object models (SOMs) and federation object models (FOMs). Two prototypes are currently in alpha release.

2. Object Model Library - A managed archive for maintaining SOMs and FOMs that is Internet-accessible. Though still under development, this component is online to support the alpha release of the OMDTs.

3. Execution Planning Tool - This tool will provide an automated means for mapping scenario-domain objects to FOM objects, provide a means of assigning modeling responsibility for scenario objects to specific federates, and specify ownership transfer conditions. This tool has not been developed.

4. Automated HLA Data Dictionary - This database will provide recommended content for FOMs and SOMs in the form of standardized, categorized data elements. The use of the Data Dictionary greatly

increases potential for object model reuse, as well as reduces misunderstanding from overloaded terminology. The HLA Data Dictionary is currently under development.

5. Data mapping support tool - This tool provides a mapping between standardized data elements in the Data Dictionary and locally defined data elements. This tool has been prototyped and is being evolved.

2.2.3 Testing Phase Components

1. Federation Testing Component - This component is used to support testing individual federates as well as an entire federation for HLA compliance. This component has not been developed.

2.2.4 Execution Phase Components

1. Runtime Monitoring Federation Component - A runtime monitoring component provides feedback during federation execution. It exists as an ancillary federate in the federation; i.e. it adds no modeling capability to the system, although it communicates with the other federates via the RTI during execution. The runtime monitor can provide information on scenario domain objects as well as collect simulation executive and network statistics. Various implementations of runtime monitors were developed by the HLA prototype federations. No further development on such a component has been initiated.
2. Runtime Management Federation Component - This component is also an ancillary federate. It is responsible for coordinating the initialization, start of execution, pausing, saving and all other federation management activity. Several implementations were developed by the HLA prototype federations. Efforts are underway to define a common management object model and to develop a runtime management tool that uses this management object model.
3. Runtime Data Collection Federation Component - This ancillary federate collects scenario domain and simulation executive domain data and records it for after-action review. The protofederations used a variety of schemes for achieving this functionality, and a prototype tool is under development.

2.2.5 Analysis Phase Components

1. Data Analysis Tool/Post Processor - This tool reduces the data logged by the Runtime Data Collection Component into useful measures of merit. The protofederations used a mix of custom tools and commercial off the shelf tools for this activity. Development has been initiated on a prototype tool.
2. VV&A Support Tool - As stated earlier, some of the tools cross the boundaries of the development phases. The VV&A activity, in particular, crosses all of the development phases. This tool supports the process of validation, verification and accreditation by automating the maintenance of traceability of the final federation execution results back to the design and original federation objectives. While no tool can automatically certify a federation, a VV&A support tool can certainly reduce the labor involved in performing VV&A, as well as guide users through a formalized process. No development has yet been initiated on a VV&A support tool.

3.0 TOOL SUPPORT STRATEGY

The modeling and simulation community, and indeed the entire software development industry, has crucial decisions to make in terms of what platforms to support, what software technologies to embrace, and what tools to invest in.

Automated software tools exist across a broad range of paradigms, from low-cost, easy-to-use desktop applications, to high-end distributed client-server environments. The correct solution for a given application involves tradeoffs between cost, maintainability, capability and ease and efficiency of use. The extreme diversity of the objectives and backgrounds of the broad spectrum of potential users of HLA makes it extremely impractical to define a single paradigm for automated tools across all federation development activities for all federations. Rather than focus our tool strategy on a particular software technology, our approach is to provide an evolutionary path to a complete, open tool architecture specification that is composed of:

1. Identification of Components
2. Interface Specification
3. Rules

Such a specification will allow tool developers to choose their own underlying technical infrastructure, be it client-server or desktop application, workstation or network based.

To develop this specification, we desire to prototype the tool architecture components as the specification is evolved, in order to verify that the specification is consistent with real-world needs and practices, in a similar manner to the way HLA itself was developed. Due to the immediate need for automated tools in the M&S user community, we identify a near term process for prototyping tools that allows the tools under development to be used now without a loss of investment in the data captured by the prototype tools when the tool architecture evolves.

Our near term prototyping process will essentially be to develop an open interface specification for critical data elements that the prototype tools will be required to support. We define a critical data element as any information that has a significant impact on either reusability of software or in any other way being able to capitalize on the efforts of previous federations.

This approach of developing an open interface specification has been implemented with the two Object Model Development Tools (OMDTs). The Object Model Library is an archive for Federation Object Models (FOMs) and Simulation Object Models (SOMs), and it maintains this archive in Object Model Template (OMT) Data Interchange Format (DIF) files. Both of the prototype OMDTs can read and write this OMT-DIF file format and, despite their very different architectures, have successfully interchanged object models via the Object Model Library.

Data elements for which DIF files are currently being developed are annotated in Figure 2, and summarized in Table 1.

Annotation	Data Element	DIF Specification
[1]	FOM, SOM	OMT DIF
[2]	Execution Details	FED DIF
[3]	Common Semantics	DD DIF

Table 1. DIF Specifications Currently Being Developed

The use of DIF files for exchanging data between tools has both advantages and disadvantages. The primary advantage is that no particular underlying technology is favored or made requisite. By requiring all tools to support a common interchange format, users can trust that an investment made in using a tool will not be lost when newer, more capable tools become available. Also, HLA users have the freedom to mix and match tools from different tool vendors as they see fit. For example, using a client-server based Execution Planning station would not prohibit a user from using a standalone desktop application for Data Analysis. Thus, our near-term support strategy satisfies our first two goals of being an open specification as well as promoting interoperability among tools. The primary disadvantage of the DIF file approach is one of efficiency. To access a portion of a particular federation's FOM, for example, the OMDT must download the entire FOM from the Object Model Library. There are no interfaces defined for automated tools to query for a subset of an object model.

As the prototype tool architecture components are used by developers of HLA federations, a more refined specification of the components will be developed by the responsible standards organization. This evolved specification will refine the definitions of the components of the architecture from those described in this paper, evolve the interface beyond the simple DIF file specifications developed during the prototyping phase, and develop a set of rules, or non-binding recommendations, for how the components interact. This evolved specification should be based on the experiences of the HLA user-community with the prototype tools, and should address issues such as security support and configuration management.

The requirement for a DIF interface will be maintained to insure that support for the efforts underway in the near term are preserved. In this way, we meet our third goal for the support plan, which is to provide the means for the tool architecture to evolve as the modeling and simulation community's needs evolve. The overall support strategy is described in the process shown in Figure 3.

4.0 SUMMARY

We have described the basic need that exists for a strategy for supporting automated tools in HLA. We established the goals for such a plan as: 1) being manifest in an open specification, 2) promoting interoperability, and 3) evolvable. We have established

as a near term strategy a process that involves assessing critical data elements, defining data interchange formats for these elements, publishing these formats and developing prototype tools. As experience with the prototype tools by the HLA user community reveals a need for more sophisticated tools and tool interfaces, the specification consisting of the tool architecture components, their interface specification and rules

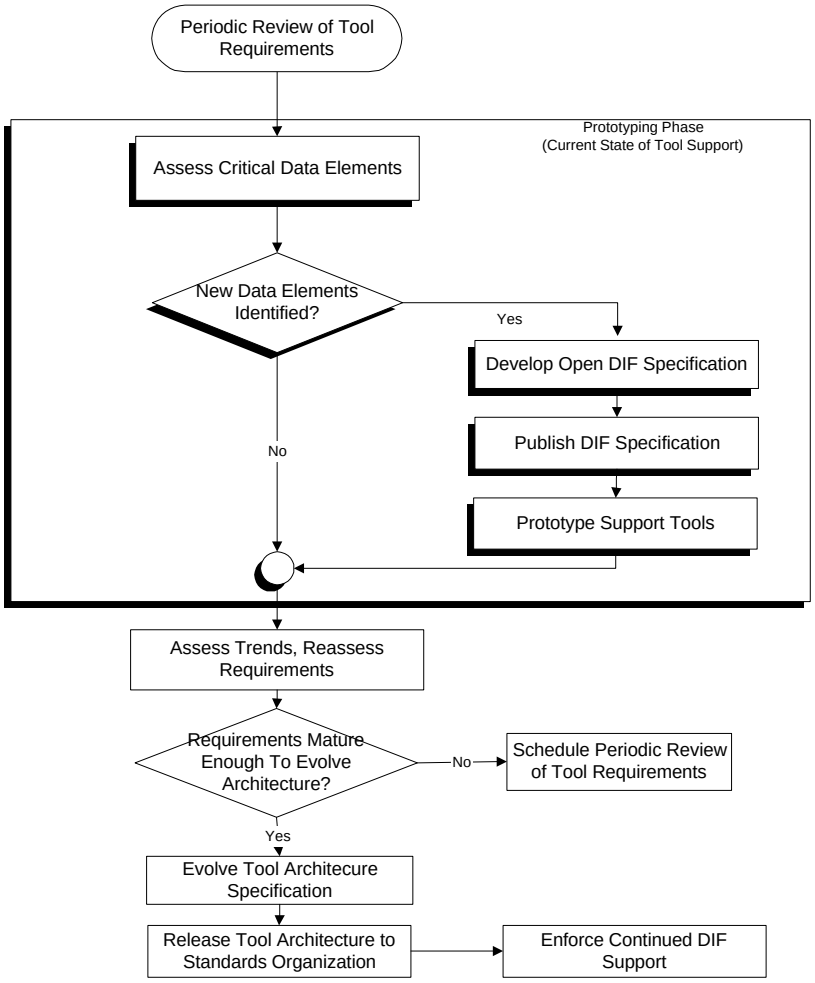

Figure 3. Tool Support Evolutionary Process

governing their use will be matured and formalized by the appropriate standards organization.

5.0 REFERENCES

[1] Defense Modeling and Simulation Office, "HLA Federation Development and Execution Process Model, Version 1.0," 21 August 1996.

[2] "The Application of Automated Tools to the HLA Federation Execution Development Process", White Paper submitted to the HLA OMT Working Group, Ken Hunt, 9 April 1996.